I used Convolutional Neural Network basically a variant of LeNet to classify German Traffic Signals.

Preprocessing Steps:

1) Convert training, validation and test from int to float32
2) Average the channels of image data as a simple mean
3) Reshape to add a dimension to training, validation and test data
4) Normalize the data set using the following formula

   (pixel - mean) / standard_deviation

Exploratory visualization:

1) Created a histogram to visualize the distribution of the various traffic signs

Model architecture:

The model followed roughly a LeNet architecture with some modification

1) Each input has 32x32 features. The input layer has dimensions [num_images, 32, 32, 1]

2) The input is fed into first convolution layer with a filter of shape [5,5,1].

   The weights for the first convolution layer are initilized through a truncated normal distribution

   with mean=0 and sigma=0.1. There are total 5*5*1*6 unique weights for this layer. The weight layer has

   dimensions [5,5,1,6]. There are 6 unique biases. This layer uses strides of dimension [1,1,1,1] and

   padding as 'VALID'. The output of this layer is of dimensions [num_images,28,28,6]. This output is passed

   through relu activation unit. Further the ouput is subjected to a dropout layer while training with a keep probability of 75%. Further this input is fed into a maxpooling layer with a kernel size of [1,2,2,1] strides of [1,2,2,1] and padding of 'VALID'. The output of this layer is [num_images, 14, 14, 6]

3) The output of the first convolution layer is fed into a second convolution layer with a filter of shape [3,3,6]. The weights for the first convolution layer are initilized through a truncated normal distribution with mean=0 and sigma=0.1. There are total 3*3*6*6 unique weights for this layer. The weight layer has dimensions [3,3,6,6]. There are 6 unique biases. This layer uses strides of dimension [1,1,1,1] and padding as 'VALID'. The output of this layer is of dimensions [num_images, 12, 12, 6]. This output is passed through relu activation unit. Further the ouput is subjected to a dropout layer while training with a keep probability of 75%. Further this input is fed into a maxpooling layer with a kernel size of [1,2,2,1] strides of [1,2,2,1] and padding of 'VALID'. The output of this layer is [num_images, 6, 6, 6]

4) The output of the second convolution layer is fed into a third convolution layer with a filter of shape [2,2,6]. The weights for the first convolution layer are initilized through a truncated normal distribution with mean=0 and sigma=0.1. There are total 2*2*6*16 unique weights for this layer. The weight layer has dimensions [2,2,6,16]. There are 16 unique biases. This layer uses strides of dimension [1,1,1,1] and padding as 'VALID'. The output of this layer is of dimensions [num_images, 5, 5, 16]. This output is passed through relu activation unit. Further the ouput is subjected to a dropout layer while training with a keep probability of 75%.

5) The output of third convolution layer is flattened to [num_images, 400]

6) The flattened layer is passed through a full layer with weights [400, 43] and 43 biases. The output of this layer is of the shape [num_images, 43]

7) The above output is converted into softmax activation unit.

Training

1) We divide the training set into batches of size 100.

2) Each batch of training input is passed into the model. We calculate the cross entropy using the output of the model and the labels converted to one hot encoding. Then we run the adam optimizer to minimize the loss with a learning rate of 0.001. Consequently

3) The whole set is passed through the model for 30 epochs.

Validation

1) The validation accuracy is calculated on passage of each epoch using the complete validation set.
2) The validation accuracy is coming to 93.3% after 25 epochs.

Test accuracy

1) The test accuracy is calculated after passage of all the epochs. We get a test accuracy of 91.1 %

5 Example images

1) We download 8 images from web related to any five of the 43 classes of the german traffic signs. We preprocess these images similar to the training set and calculate the accuracy over them by passing through the model. We get an accuracy of 87.5% i.e 7 of the 8 images were correctly classified. The 7 out of 8 images were classified with 100% probability.

2) For each of these 5 images we identify the top five probabilities.

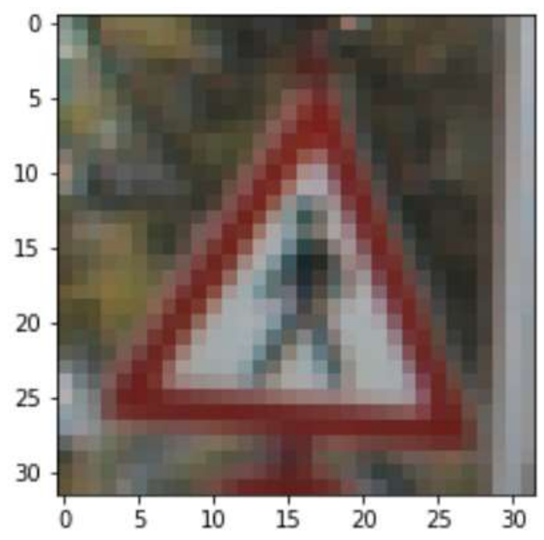
Image	**Prediction**
Speed limit (60km/h)	Speed limit (60km/h)
Roundabout Mandatory	Roundabout Mandatory
Yield	Yield
Children crossing	Children crossing
Double curve	Double curve
Pedestrians	Right-of-way at the next intersection
Traffic signals	Traffic signals
Wild animals crossing	Wild animals crossing

Top 5 probabilities for the 8 images

The 7 out of 8 images were classified with 100% probability.

```
[1.000000e+00, 0.000000e+00, 0.000000e+00, 0.000000e+00,
 0.000000e+00]

[1.000000e+00, 0.000000e+00, 0.000000e+00, 0.000000e+00,
 0.000000e+00]

[1.000000e+00, 0.000000e+00, 0.000000e+00, 0.000000e+00,
 0.000000e+00]

[1.000000e+00, 0.000000e+00, 0.000000e+00, 0.000000e+00,
 0.000000e+00]

[1.000000e+00, 0.000000e+00, 0.000000e+00, 0.000000e+00,
 0.000000e+00]

[1.000000e+00, 0.000000e+00, 0.000000e+00, 0.000000e+00,
 0.000000e+00]

[1.000000e+00, 7.938287e-22, 0.000000e+00, 0.000000e+00,
 0.000000e+00]

[1.000000e+00, 0.000000e+00, 0.000000e+00, 0.000000e+00,
 0.000000e+00]
```

The pedestrian image was not classified properly because the training set has some jittery, low resolution images. Below is an example image from the training set for the pedestrian sign.


0
5
10
15
20
25
30
0
5
10
15
20
25
30